GlideCursor: Pointing with an Inertial Cursor

Michel Beaudouin-Lafon, Stéphane Huot, Halla Olafsdottir, Pierre Dragicevic

GlideCursor: Pointing with an Inertial Cursor

Michel Beaudouin-Lafon[1,2] Stéphane Huot[1,2] Halla Olafsdottir[1,2] Pierre Dragicevic[2]

[1]Univ Paris-Sud & CNRS (LRI)
F-91405 Orsay, France

[2]INRIA
F-91405 Orsay, France

ABSTRACT

Pointing on large displays with an indirect, relative pointing device such as a touchpad often requires clutching. This article introduces *gliding*, where the cursor continues to move during the clutching gestures. The effect is that of controlling the cursor as a detached object that can be pushed, with inertia and friction similar to a puck being pushed on a table. We analyze gliding from a practical and a theoretical perspective and report on two studies. The first controlled experiment establishes that gliding reduces clutching and can improve pointing performance for large distances. We introduce *cursor efficiency* to capture the effects of gliding on clutching. The second experiment demonstrates that participants use gliding even when an efficient acceleration function lets them perform the task without it, without degrading performance.

Keywords Pointing; Clutching; Gliding; Inertia; Friction; Cursor Efficiency; Wall-size display

General Terms Design; Human Factors

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation - User Interfaces**]: Input devices and strategies

1. INTRODUCTION

Controlling a cursor on a large display with a mouse or touchpad, e.g. to navigate large images or to select objects [8], often requires *clutching*: The small size of the input area relative to the display surface forces users to recalibrate their hand or finger when they reach its border. Casiez et al. [12] report that up to 25% of the time can be spent clutching on a large wall-size display controlled with a mouse. Clutching degrades pointing performance for two reasons: The cursor does not move while clutching, and the cursor velocity cannot follow the typical bell curve that characterizes an optimal aimed movement [25] but instead is limited by the maximum velocity that can be achieved through the limited input area. As a result, clutching breaks Fitts' Law [12]: movement time, instead of being proportional to the index of difficulty of the task, becomes primarily proportional to the distance to the target.

M. Beaudouin-Lafon, S. Huot, H. Olafsdottir, P. Dragicevic.
GlideCursor: Pointing with an Inertial Cursor.
In AVI'14 : Proceedings of the International Working Conference on Advanced Visual Interfaces, ACM, May 2014.
Authors Version

Figure 1: Rolling a hoop with intermittent control.

The standard approach to reduce clutching is to increase "mouse acceleration", the transfer function that maps fast input movements to larger cursor motion [12]. However, the transfer function becomes very sensitive to changes in the speed of input movements when trying to combine speed and precision for large displays and small input surfaces, making it difficult to control the cursor [26].

If we cannot get rid of clutching, can we take advantage of it to improve pointing performance? At the end of a targeted movement, users slow down their movement to zero velocity before selecting the target. But when clutching, users typically perform flicking gestures with high velocity, including at lift-off, when the finger or mouse is lifted from the surface. High velocity at lift-off therefore characterizes the user's intention to continue the aimed movement.

In the real world, flicking gestures are often used to throw an object, for example a puck on a flat surface. Throwing, however, is imprecise: once released, the object moves according to the laws of physics (inertia and friction), and one often has to wait until it stops to acquire it again. But intermittent control is also possibe to correct the course of a moving object, such as a hockey player pushing the puck or a child rolling a hoop with a stick (Fig. 1).

Based on these observations we introduce GlideCursor, a technique that applies inertia to the cursor so that it *glides* when thrown by the user. When clutching, the user's flicking gestures throw the cursor, which continues its course, but the user can instantly regain control by putting down the mouse or touching the trackpad.

In order to implement GlideCursor, we need to measure the velocity of the cursor at lift-off. Commercial mice do not sense when they are being lifted[1] but on a touch sensitive input surface, such as a trackpad, we do know when the user's fingers are in contact. For this reason, the rest of this paper considers the GlideCursor to be controlled by a trackpad or other touch-sensitive surface.

After a review of related work, we describe our implementation of the gliding cursor and perform a theoretical analysis of its performance. We then report on two studies conducted to assess GlideCursor and conclude with future work.

[1]See, however, Casiez et al's instrumented mouse [12].

2. RELATED WORK

A form of gliding called Momentum is available in Synaptics's *Gesture Suite for TouchPads*[2]. However, to the best of our knowledge, no formal study of the potential benefits of this technique has been published. In the large body of literature on pointing facilitation, we focus on two categories of techniques: those that involve throwing, tossing and flicking, and those that can reduce clutching.

2.1 Throwing, tossing and flicking

A number of interfaces let users throw and toss objects. For example, document icons can be tossed around in the BumpTop [1] desktop interface. Throwing and tossing are often triggered by a flicking gesture, i.e. a rapid movement of the mouse, pen or finger [17]. Superflick [27] uses flicking to throw objects on a table, while Multi-flick [2] lets users navigate a long document. Scrolling is the most well-known application of inertia and friction in user interfaces, popularized by the inertial scrolling of smartphones and tablets that lets users scan a long document by flicking [3].

Some techniques only use part of the physical metaphor: Push-and-throw [20] is a drag-and-drop technique where the user can throw an object towards a target, but the throw is not initiated with a throwing gesture. Boomerang [22] uses a throwing gesture to suspend a drag-and-drop operation. The throwing gesture releases the object being dragged, but without autonomous motion.

Despite a number of techniques that use throwing, tossing and flicking in user interfaces, we are not aware of previous work where they were applied to the cursor itself. In addition, with the important exceptions of inertial scrolling and flick-and-brake [5], users lose control of the object after throwing it or have to re-acquire it. By contrast, GlideCursor is constantly under user control: simply touching the trackpad stops the cursor if it is moving.

2.2 Pointing techniques that reduce clutching

Pointing facilitation techniques can be classified along three non-independent dimensions:

Target-aware vs. *target-agnostic*, i.e. techniques that take into account the locations and sizes of the potential targets on the display or not. In this work we focus on target-agnostic techniques because they are more generally applicable, although GlideCursor is compatible with target-aware techniques such as Semantic Pointing [9], Bubble Cursor [18] or Dynaspot [14].

Direct vs. *indirect*, i.e. techniques where the input and display area are the same surface or not. GlideCursor is an indirect technique. It could be adapted to a direct-touch surface but the cursor would be decoupled from the contact point.

Absolute vs. *relative*, i.e. techniques that map a position vs. a displacement sensed by the input device into a position or displacement of the cursor. GlideCursor is inherently a relative technique, because the position of the cursor must be decoupled from that of the input device. As such it can benefit from a *transfer function* between input and cursor displacements.

Pointing facilitation techniques typically try to reduce distance to target, increase target width, or both [6]. Since clutching is caused primarily by the increased distance to target, we focus on techniques that reduce that distance.

Drag-and-pop [7] detects the direction of motion of a drag operation and brings the potential targets in that direction close to the cursor. Semantic Pointing [9] and Object Pointing [19] shrink or remove the space between targets. Bubble Cursor [18] fills out the non-target space, resulting in smaller distances. While all these techniques are target-aware, GlideCursor is target-agnostic.

MAGIC [30] is a target-agnostic technique that reduces distance by tracking the user gaze to anticipate the movement of the hand, while Delphian Desktop [4] predicts the distance to target from the velocity profile of the input movement. However the former requires an eye-tracker and the latter suffers from the lack of information in the early phase of a pointing movement to accurately infer its magnitude [25]. GlideCursor corrects this problem by letting users regain control after the cursor has been thrown.

Transfer functions are widely used to overcome the mismatch between input and display size. Casiez et al. [12] derive formulas to assess the range of gains available for a given input-output configuration, and recommend using non-constant transfer functions to combine speed and precision. Nancel et al. [26] present transfer functions to control a cursor on a wall-size display with a tablet and even a smartphone without the need for clutching, but they can be very sensitive to small changes in velocity. As a result, users tend to clutch even though the transfer function lets them avoid clutching.

Clutching can also be avoided by using *rate control* instead of position control, i.e. mapping the input device's position to the cursor's speed. However, rate control requires elastic or isometric devices such as joysticks and is hard to use [10, 15]: at low gains it takes time to reach distant targets, while at high gains corrective movements are difficult [11]. RubberEdge [13] combines position and rate control, but requires specialized hardware and only outperforms regular pointing when clutching three or more times.

In summary, many techniques that reduce clutching are either target-aware or require special hardware (eye tracker, input device). The remaining techniques manipulate the transfer function or extract information from the early phases of the movement, both of which scale poorly with amplitude.

2.3 Movement Efficiency in Pointing Tasks

Studies of pointing often focus on time efficiency measures, e.g, average completion times or index of performance [28], but ignore movement efficiency. Movement efficiency is linked to subjective factors such as fatigue, comfort and effort that are typically assessed indirectly with questionnaires.

Several objective measures of movement efficiency consider a movement to be efficient when it minimizes the distance covered, i.e. when it is close to a straight path: Zhai and Milgram [29] define efficiency (or coordination) in multi-DOF movements as the percent difference between the length of the path traveled and the shortest path; Mackenzie et al [23] define six metrics that capture how close a pointing movement is to an ideal movement. The only objective measure of movement efficiency we know of that captures clutching is the time spent clutching as a percentage of task completion time [12]. We define a related metric called *cursor efficiency* that also accounts for gliding.

3. ADDING INERTIA TO THE CURSOR

We now describe GlideCursor in detail. We assume that the cursor is controlled through a touch-sensitive surface, typically a trackpad, with a single finger. We track the velocity of the finger while it is in contact with the trackpad. If the velocity at lift-off is larger than a given threshold, called minimal trigger velocity, we initiate gliding. An animation loop moves the cursor according to a motion law (see below) and stops the animation when any of the following conditions occurs:

- The velocity of the cursor is below a threshold;
- The cursor hits one of the borders of the display surface;
- The user touches the input surface again.

[2] http://www.synaptics.com/solutions/technology/touchpad

[3] Inertial scrolling dates back to 1984 at least: http://www.folklore.org/StoryView.py?project=Macintosh&story=Thunderscan.txt.

We empirically set the minimal trigger velocity to 1000 pixel/s on our 5.5 meter, 20480 pixel-wide wall-size display.

The first condition corresponds to the cursor stopping by itself. Since some motion laws decrease velocity without ever actually reaching zero, we avoid useless computation by stopping the cursor when its motion becomes negligible.

The second condition ensures that the cursor stays on screen. We could also implement bouncing (with an elastic or non-elastic shock) or wrapping around the display, as in TorusDesktop [21]. Our current solution is consistent with the normal behavior of the cursor and takes advantage of the fact that targets on the borders of the screen are easier to reach [3].

The third condition lets the user regain control of the cursor to adjust its course by performing either another flicking gesture that results in more gliding, or a terminal pointing gesture. If the target is large enough, such as a window, the user can simply tap the trackpad to stop the cursor and select the underlying window.

Some versions of inertial scrolling, e.g., on Apple iOS, let the user accelerate the motion by performing successive flicks. We have not implemented this behavior because unlike scrolling a very long document, pointing on a large display does not let users take advantage of this feature. Also, accelerated scrolling does not let users control deceleration: every new flick accelerates the motion even if it is performed slowly, resulting in a sense of loss of control.

3.1 Motion laws

The physical model of GlideCursor is that of pushing a puck on a table. Two laws of physics are involved: inertia, which continues to move the puck at its current speed when released; and friction, which progressively reduces its speed. Friction is a complex physical phenomenon. Consistent with a previous analysis of friction in tossing tasks [24], we consider three types of friction:

- *Static friction* holds two surfaces in contact without sliding, such as a puck that stays steady on a slanted surface. It is characterized by a coefficient of static friction, μ_s, such that the object does not move as long as the lateral force applied to it is less than $F = \mu_s F_n$, where F_n is the normal reaction at the contact point (typically the opposite of the weight).
- *Kinetic friction* resists the lateral movements of two surfaces in contact, such as a puck sliding on a slanted table. It is characterized by a coefficient of kinetic friction, μ_k, such that the friction force that resists lateral motion is $F = \mu_k F_n$, where F_n is defined as above.
- *Fluid friction* resists the movements within a fluid, such as stirring a spoon in a pot of liquid honey. It is characterized by a coefficient of viscosity, μ_f, such that the friction force is proportional to the speed of motion: $F = -\mu_f v$ where v is the velocity of the moving object.

Our model ignores static friction, although it is implemented indirectly through the velocity threshold below which the motion stops. Since the coefficient of static friction is higher than that of kinetic friction for most materials, an accurate model would require the user to "push" the cursor harder to get it to move, creating an unpleasant user experience.

Kinetic friction seems a more appropriate model than fluid friction, however our informal comparison of the two led to a slight preference for fluid friction. This can be explained by analyzing the equations of motion for an object of mass m on a flat surface, i.e. for which $F_n = -mg$ (g is gravity).

With kinetic friction, $F = -\mu_k mg = ma$ (a is the acceleration). Therefore $a = -\mu_k g$, the velocity $v = v_0 - \mu_k gt$ (v_0 is the initial velocity) and the position p along the moving direction is $p = v_0 t - \frac{1}{2}\mu_k g t^2$. The object stops when its velocity reaches 0,

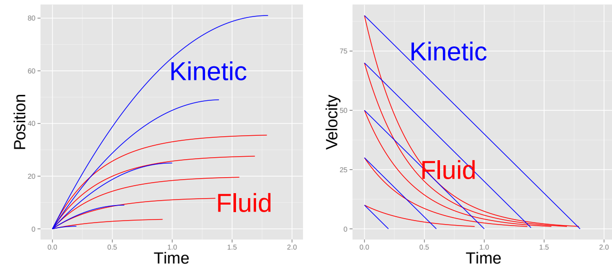

Figure 2: Plot of position (left) and velocity (right) over time for an object under kinetic friction (in red) and fluid friction (in blue) on a flat plane, for five values of the initial velocity.

i.e. at time T such that $v_0 = \mu_k g T$, or $T = v_0/\mu_k g$. At this time the distance covered is $D = v_0^2/2\mu_k g$.

With fluid friction, $F = -\mu_f v = ma$. Therefore $a = -\frac{\mu_f}{m} v$, $v = v_0 e^{-(\mu_f/m)t}$ and $p = \frac{m}{\mu_f} v_0 (1 - e^{-(\mu_f/m)t})$. Although the velocity never reaches zero, the maximum distance covered by the object is $D = \frac{m}{\mu_f} v_0$. If we set a minimum velocity v_{min} below which we consider the object stopped, it is reached at time $T = \frac{m}{\mu_f} log(\frac{v_0}{v_{min}})$.

Figure 2 plots position and velocity for five values of v_0 for both kinetic and fluid motion. Interestingly, the visual aspect of these curves does not match the perception of the corresponding motion when applied to the cursor: while kinetic friction causes velocity to decrease linearly, the perception is that of a cursor moving fairly steadily and slowing down abruptly at the end of its course. Conversely, fluid friction is perceived as a steady decrease in velocity, even though the curves show a non-linear decrease. This may be because in the real world we seldom experience kinetic friction on a flat surface[4] but rather on a slope when an object starts sliding, in which case its motion is accelerated by its weight, not decelerated.

Figure 2 also shows that kinetic friction is faster than fluid friction: it goes further in the same amount of time. It is also more sensitive to initial velocity since the range of final distances and times is much larger than for fluid friction. Experiment 2 below compares the effects of these two types of friction.

Note that other laws could be applied. While gravity or wind may be entertaining but not very useful, distortion fields based on the content of the screen could make important objects attract the cursor and dangerous ones repel it, as in Semantic Pointing [9]. This is left for future work.

3.2 Theoretical analysis: Cursor Efficiency

Gliding is meant to facilitate pointing when the user needs to clutch to reach a distant target. Casiez et al. [13] have introduced the only model we know for clutching, which we rewrite as follows (we assume an identity transfer function):

$$MT = 2.N.T_c + a + b.log_2\,(D_e/W + 1) \quad (1)$$
$$\text{with } N = floor(D/d_e) \text{ and } D_e = D - N.d_e$$

N is the number of clutches, T_c the clutching time, d_e the operating range of the input device, and a, b, D and W the usual Fitts' Law parameters. D_e is the distance left to cover when no more clutching is needed ($D_e < d_e$) and defines the terminal pointing phase. Since N is the only term that increases monotonically with D, the linear term is bound to dominate the log term as D increases.

[4] Ice skating or playing curling are exceptions; playing ball or pool involves other physical phenomena because the balls are rolling.

What happens when we introduce gliding? Since the cursor continues to move while the user is clutching, the term d_e in the above formulas must be replaced with $d_e + d_g$ where d_g is the distance covered by the cursor while gliding. This should result in fewer clutches (lower N) and a shorter pointing time. However, this gain may be offset by two phenomena.

First, the autonomous motion of the cursor is likely to increase the error, requiring more and/or longer corrective motions when closing up on the target. Typical errors include overshoot, where the cursor goes beyond the target, and aiming errors, where the cursor drifts orthogonally to the direction of the target.

Second, the user may take advantage of gliding to be more lazy, since a small flicking gesture now has the same effect as a longer tracking motion with a classic cursor. This reduces the operating range d_e but not necessarily the gliding distance d_g since it depends only on velocity at lift off: a short but fast flicking gesture may result in higher velocity and therefore longer gliding distance. However, variability increases with movement amplitude [24], making it more risky to throw the cursor at high speed with a quick flicking gesture. Instead, similar to a child using small touches to adjust the trajectory of a hoop, users are likely to take advantage of gliding by successive corrective motions rather than long throws.

In summary, the gain in movement time may not be the primary advantage of GlideCursor. We are more likely to observe a qualitative change in the way users control the cursor, where absolute performance at the expense of fast and tiring clutching movements is traded off for good-enough performance at a fraction of the effort. This theory is consistent with the models that describe aimed movements as a succession of initial open-loop, ballistic movements, followed by corrective closed-loop movements [25].

Cursor Efficiency.

Traditional measures of pointing performance include *Movement Time* (MT), *Index of Difficulty*, $ID = log_2(\frac{D}{W} + 1)$, measured in bits, and *Index of Performance* (IP) in bits/s, usually computed as the inverse of the slope of the linear regression of movement time as a function of the index of difficulty ($MT = a + ID/IP$).

While IP is an objective measure of the rate of information transfer from user to system, it is a holistic measure that does not account for discontinuous control, as with clutching, or autonomous cursor motion, as with gliding. It also assumes that movement time follows Fitts' Law, which is unlikely in these situations.

When clutching, part of the movement time is lost recalibrating the hand. Let us call it *lost time*, LT, and let us call *User Time*, UT, the time during which the user controls the cursor. We have $MT = UT + LT$. The metric introduced by Casiez et al. [13] is the ratio of clutch time to movement time: $r = LT/MT$. When gliding, the time when the cursor glides is neither part of UT nor LT. Let us introduce *Gliding Time*, GT, the total time during which the cursor moves by itself. Then $MT = UT + GT + LT$ and r can be rewritten as $r = 1 - (UT + GT)/MT$.

We define *Cursor Efficiency*, $CE = (UT + GT)/MT$, as the amount of time the cursor is in motion relative to total task time, i.e. until the target is acquired. In the absence of clutching (and therefore gliding), $CE = 1$, the optimal value. In the presence of clutching but without gliding, CE is lower than 1: Casiez et al. [12] found clutching time to be up to 25% of movement time, i.e. $CE = .75$. With gliding, CE should be higher because gliding time overlaps clutching time. It may even reach 1 if the user can adjust the cursor before it stops, like the child pushing the hoop.

We now assess this theoretical analysis with Experiment 1 to test the hypothesis that gliding increases Cursor Efficiency without sacrificing raw performance, i.e. overall movement time.

4. EXP. 1: THE GLIDING EFFECT

In this experiment, we assess the effect of gliding on large amplitude pointing tasks performed using a touch-sensitive tablet in front of a large wall-size display. We use a standard reciprocal pointing task [16] with IDs between 4 and 8 bits. Even though we only test horizontal movements, we use circular targets and let the cursor move in both dimensions. This increases external validity because throwing the cursor amplifies aiming errors.

To establish the effects of gliding, we compare two extreme levels of friction with an intermediate one. The first extreme case is infinite friction, which causes the cursor to never move by itself. This is *Classic* pointing, our baseline. The other extreme case, called *Flying*, is zero friction, which corresponds to pure inertia: the cursor keeps its initial release velocity until stopped by the user or the screen border. The third level, *Gliding*, is an intermediate value calibrated through pilot testing to ensure that users need some clutching with all of the target distances used in the experiment.

Our theoretical analysis leads us to the following hypotheses:

H1: Movement time does not follow Fitts' Law;

H2: Inertia does not degrade movement time;

H3: Inertia reduces the number of clutches;

H4: *Cursor Efficiency* increases when friction decreases: *Classic* is lowest, then *Gliding*, *Flying* is highest.

Participants.

12 right-handed adults (6 male), aged 23–39, all heavy computer users and familiar with touch devices, participated.

Apparatus.

The software is implemented in Java and runs on a 5.5×1.8m wall-size display [8], with an overall resolution of 20480×16000 (8×4 30" Apple Cinema monitors, 101ppi). It is powered by a cluster of 16 Apple Mac Pro computers and an Apple MacBook Pro front-end running the main application. A dedicated Apple iOS application runs on an iPad 3 tablet to capture the user touches and send them to the front-end. The control-display ratio between user input and cursor movement is constant and set to 5. Moving the finger 1mm on the tablet moves the cursor 20 pixels on the screen.

Procedure.

The [3x3x2] within-subject design has the following factors:

- TECHNIQUE, with three treatments:

  Classic: the traditional cursor that only moves when touching the tablet, corresponding to infinite friction;

  Gliding: an inertial cursor that continues to move after lift-off according to its initial velocity and the coefficient of friction. We use kinetic friction with a coefficient of 150 and a minimal trigger velocity of 1000 pixel/s.

  Flying: gliding with zero friction, i.e. the cursor moves at constant speed until it reaches the border of the display or the user touches the tablet;
- D, the distance to target, with three levels: *D1* (4500 pixels or 112.4cm), *D2* (9000 pixels or 224.9cm) and *D3* (18000 pixels or 449.8cm);
- W, the width of the target (i.e. its diameter), with two levels: *W1* (40 pixels or 10mm) and *W2* (160 pixels or 40mm).

Participants stand about 2m away from the wall-size display and are instructed to perform the task as quickly as possible. The tablet is held on their non-dominant arm and they interact using their dominant hand. The task is a reciprocal pointing task with a series of 7 consecutive trials for each combination of D×W, the first of which is ignored in the analyses. The first target appears on the left of the display and the cursor at its center. Targets are selected by moving the cursor inside the target and lifting the finger off the tablet. As a result, a trial cannot end with a selection error. After a successful selection, the next target appears immediately on the opposite side of the display. We use selection on lift-off to reproduce Fitts' reciprocal pointing task. We did not observe any spurious gliding when tapping because of the velocity threshold and the small hysteresis we introduce at lift-off.

Trials are grouped by TECHNIQUE, with two blocks per group, and the order of TECHNIQUE is counterbalanced across participants. The order of D×W conditions is randomized in each block. Before each new TECHNIQUE, participants perform a training block with four series of trials (W=*W2*, all D values). Sessions last about 30 minutes and participants answer a short questionnaire at the end.

Data collection.

We collected four measures for the 3 TECHNIQUE × 3 D × 2 W × 2*6 TRIALS × 12 PARTICIPANT = 2592 measured trials:

- MT**:** from initial cursor movement to selection of the target;
- USER TIME**:** time spent controlling the cursor;
- GLIDING TIME**:** time spent while the cursor is gliding; and
- CLUTCHES**:** number of clutches.

4.1 Results

A Shapiro-Wilk test[5] confirms that the distribution of MT is normal. A Grubb's test detects fewer than 2% outliers, which we keep. All analyses are therefore performed on the complete dataset.

Movement time and Fitts' Law.

We first test whether MT follows Fitts' Law. We aggregate MT by index of difficulty (ID) for each PARTICIPANT and TECHNIQUE. A bivariate linear fit of the mean MT by ID for each TECHNIQUE gives low r^2 values by Fitts' Law standards, and large negative intercepts (*Classic*: $a = -1.48$, $b = 0.65$, $r^2 = .84$; *Gliding*: $a = -0.93$, $b = 0.55$, $r^2 = .86$; *Flying*: $a = -0.16$, $b = 0.44$, $r^2 = .76$). These results support **H1** and are consistent with Casiez et al. [13].

We then test Casiez et al.'s model of clutching (Eq. 1). We extract the mean effective operating range of the device from our data (141mm) and compute the adjusted index of difficulty for each trial. As expected, the fit is poor for *Gliding* ($r^2 = .56$) and *Flying* ($r^2 = .20$). More surprisingly, it is worse than Fitts' Law for *Classic* ($r^2 = .71$). Since neither Fitts' Law nor this model apply to gliding and flying, we only analyze the effects of D and W.

Effect of inertia on movement time.

An ANOVA on MT with the model TECHNIQUE×D×W×RANDOM (PARTICIPANT) reveals no significant effect of TECHNIQUE but significant effects of D ($F_{2,22} = 343.04, p < 0.0001$) and W ($F_{1,11} = 1305.07, p < 0.0001$), as well as significant D×TECHNIQUE ($F_{4,44} = 22.97, p < 0.0001$) and W×TECHNIQUE ($F_{2,22} = 3.99, p = 0.0331$) interaction effects. For D, a post-hoc Tukey HSD test shows that all levels are significantly different (avg. *D1*: 2.2s, *D2*: 2.77s and *D3*: 3.77s). For W, a Student's t-test shows a significant difference between *W1* (3.3s) and *W2* (2.53s).

The D×TECHNIQUE interaction reveals the impact of inertia on MT (Fig. 3): The techniques are not significantly different for the small and medium distances, but a post-hoc Tukey HSD test shows that *Flying* (avg. 3.5s) and *Gliding* (avg. 3.69s) are significantly faster than *Classic* (avg. 4.12s) for the larger distance. In contrast, the W×TECHNIQUE interaction does not reveal a strong effect of TECHNIQUE on W: the difference is significant only for the smaller width and between *Flying* (avg. 3.22s) and *Classic* (avg. 3.53s).

These results support **H2** and suggest that inertia can in fact save substantial time in large amplitude tasks: up to 10% for *Gliding* and 15% for *Flying* compared to *Classic*.

[5] All analyses are performed with the SAS JMP statistical platform.

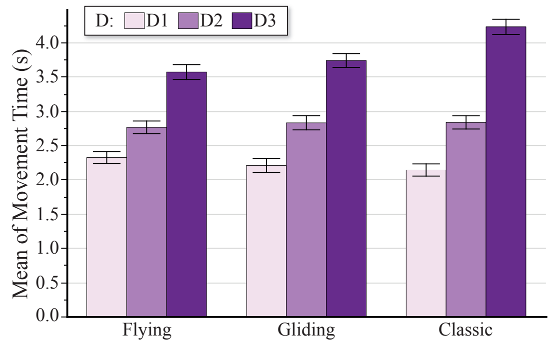

Figure 3: Mean of MT by D and TECHNIQUE. *Flying* and *Gliding* are faster than *Classic* for larger distances.

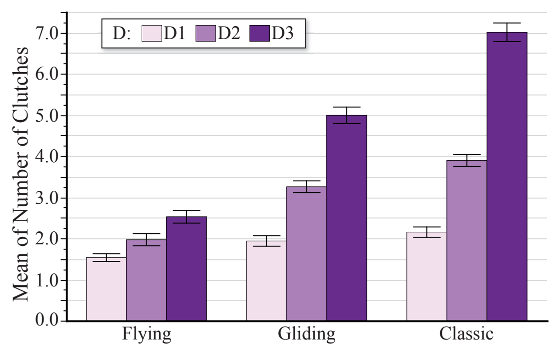

Figure 4: Mean of CLUTCHES by D and TECHNIQUE. *Gliding* and *Flying* reduce the number of clutches as distance increases.

Number of clutches.

The same analysis as above on number of CLUTCHES reveals similar effects of D ($F_{2,22} = 291.42, p < 0.0001$) and W ($F_{1,11} = 80.05, p < 0.0001$), with significant D×TECHNIQUE and W× TECHNIQUE interaction effects (resp. $F_{4,44} = 55.49, p < 0.0001$ and $F_{2,22} = 5.37, p = 0.0126$). TECHNIQUE also has a sigificant effect on CLUTCHES ($F_{2,22} = 43.38, p < 0.0001$). Figure 4 shows that the number of CLUTCHES increases with D much faster for *Classic* than for *Flying*, with *Gliding* in between. For the small and medium distances, the only significant difference is between *Classic* (avg. 2.12 & 3.85) and *Flying* (avg. 1.5 & 1.96), while for the large distances all differences are significant (avg. *Classic*: 6.89, *Gliding*: 4.97, *Flying*: 2.47). Regarding W, the number of CLUTCHES is slightly higher for small targets for all techniques, probably because small targets require more adjustment movements to be selected.

These results support **H3**, i.e. that inertia significantly reduces the number of clutches. We also note that mean clutch time increases with gliding while actual operating range decreases (*Classic*: 218ms/108mm, *Gliding*: 275ms/93mm, *Flying*: 469ms/47mm).

Cursor Efficiency.

Cursor efficiency, CE, is the ratio of USER TIME+GLIDING TIME to MT, i.e. the total time during which the cursor is moving divided by the time to reach the target. CE is between 0 and 1, with higher values indicating a more efficient use of movement time.

An ANOVA on CE with the model TECHNIQUE×D×W×RANDOM (PARTICIPANT) reveals significant effects of TECHNIQUE ($F_{2,22} = 194.67, p < 0.0001$), D ($F_{2,22} = 141.85, p < 0.0001$) and W ($F_{1,11} = 13.05, p = 0.0041$), and significant D×TECHNIQUE

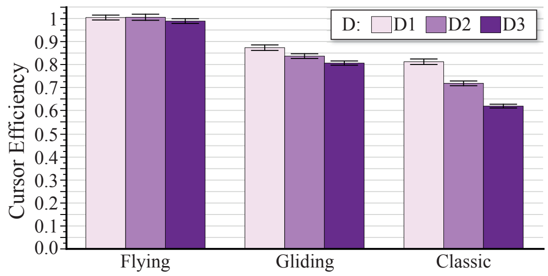

Figure 5: Mean of CE by D and Technique. *Flying* is almost optimal and *Gliding* is more efficient than *Classic*.

($F_{4,44} = 42.81, p < 0.0001$) and W×Technique ($F_{2,22} = 17.60$, $p < 0.0001$) interaction effects. Post-hoc Tukey HSD tests show that all differences are significant: Technique (avg. *Flying*: 0.998, *Gliding*: 0.84, *Classic*: 0.76), D (avg. *D1*: 0.89, *D2*: 0.85, *D3*: 0.8) and W (post-hoc Student's test, avg. *W1*: 0.86, *W2*: 0.84).

Figure 5 shows the D×Technique interaction effect. An anova on CE for the model D×W×Random(Participant) by Technique confirms that D has no effect for *Flying* but does for *Gliding* ($F_{2,22} = 20.99, p < 0.0001$) and *Classic* ($F_{2,22} = 173.98, p < 0.0001$).

These results strongly support **H4**: inertia increases cursor efficiency, which is consistently very close to 1 for *Flying*.

Qualitative results.

The questionnaire results are consistent with the performance results. 9 out of 12 participants rated *Flying* as the fastest technique, 1 rated *Gliding* as fastest and 1 *Flying/Gliding*. The last one did not know. 8 preferred *Flying*, 3 preferred *Gliding*, 1 preferred *Flying/Gliding*. A participant mentioned that *Gliding* was a bit "sticky" and would have worked better with less friction.

Three participants commented that left-to-right *Flying*/*Gliding* seemed faster and had higher angular precision. Our analyses did not show evidence of this effect, which should be studied further.

5. EXP. 2: GLIDING WITH ACCELERATION

Experiment 1 established the effect of gliding in a pointing task with constant gain. However variable gain or "acceleration" is often used when controlling a cursor with a relative device. How does gliding perform in the presence of acceleration? To answer this question, we ran a second experiment where the cursor acceleration lets users perform the pointing task *without* clutching. Our goal is to see how participants trade acceleration for gliding, and its impact on performance. Our three hypotheses are:

H1: *Gliding* does not significantly affect MT;
H2: *Gliding* is used even if the task does not require clutching;
H3: *Gliding* increases the *Cursor Efficiency*.

Participants.

12 adults (11 from Exp. 1, 6 male), aged 23–40, participated.

Apparatus.

Hardware and software are the same as Experiment 1. The transfer function is replaced by the acceleration function from Nancel et al. [26], which was designed for wall-size displays. The parameters are set so that participants can traverse the entire display without clutching by moving their finger swiftly from one side of the tablet to the other: $V_{min} = 0.006m/s$, $V_{max} = 0.37m/s$, $CD_{min} = 0.22$, $CD_{max} = 43.1$, $\lambda = .02s/mm$, $ratio_{Inf} = 0.5$.

Procedure.

The [3x3x2] within-subject design has the same factors as Exp. 1:

- Technique, with three new treatments:
  Acceleration: regular cursor with the above transfer function
  Gliding Kinetic: inertial cursor with kinetic friction *and* the above acceleration. The coefficient of friction is set to 55 so that participants can cross the entire display with one flicking gesture (minimal trigger velocity is 1000 px/s). Since gliding is now combined with a transfer function that can produce high velocities, we bound the initial gliding velocity to 40 000 px/s;
  Gliding Fluid: same as above with fluid friction. To get a similar gliding distance, the coefficient of friction is set to 5.5 and stopping velocity to 200 px/s.
- D, the distance to target, with the same three levels: *D1* (4500 pixels or 112.4cm), *D2* (9000 pixels or 224.9cm) and *D3* (18000 pixels or 449.8cm);
- W, the width of the target, with the same two levels: *W1* (40 pixels or 10mm) and *W2* (160 pixels or 40mm).

We obtain a consistent baseline by presenting *Acceleration* first, followed by the two gliding conditions, in counterbalanced order across participants. This design lets participants get used to the steep slope of the transfer function, which generates high velocities. Also, by conditioning participants to use acceleration, we bias them *against* the use of gliding in the subsequent conditions, since the same acceleration is present in all three conditions and the task can always be performed without clutching and therefore without gliding. Thus, if participants use gliding, it indicates a conscious choice on their part. To make this clear, the instructions explicitly state that they do *not* have to use gliding, but that they should perform the task as quickly and comfortably as possible, as if they were going to do it for an extended period of time.

The rest of the design is identical to Exp. 1. Sessions last 20 minutes and participants are given a short questionnaire at the end.

Data collection.

We collected the same measures as in Experiment 1 for the 3 Technique × 3 D × 2 W × 2*6 Trials × 12 Participant = 2592 trials: MT, User time, Gliding time and Clutches.

5.1 Results

As with Experiment 1, we check the collected data for normality and outliers, and keep all trials for our analyses.

Movement time.

An anova on MT with the model Technique×D×W×Random (Participant) reveals no significant effect of Technique but significant effects of D ($F_{2,22} = 107.40, p < 0.0001$) and W ($F_{1,11} = 223.23, p < 0.0001$), with a significant Technique×D×W interaction ($F_{4,44} = 2.73, p = 0.0406$). For D, a post-hoc Tukey HSD test shows all differences to be significant (avg. *D1*: 2.02s, *D2*: 2.24s and *D3*: 2.71s). For W, a Student's t-test shows a significant difference (avg. *W1*: 2s, *W2*: 2.64s). In summary, overall performance is similar across Technique, supporting **H1** (Fig. 6).

Clutching.

An anova on the number of Clutches with the model Technique×D×W×Random(Participant) reveals significant effects of D ($F_{2,22} = 109.66, p < 0.0001$) and W ($F_{1,11} = 26.33, p = 0.0003$), but no interaction effects. For D (Fig. 7), a post-hoc Tukey HSD test shows significant differences between all three levels (avg. *D1*: 0.95, *D2*: 1.56, *D3*: 2.39). For W, a Student's t-test shows a significant difference (avg. *W1*: 1.78, *W2*:1.48).

The stability across techniques can mean either that participants behave the same in all three conditions, ignoring gliding and us-

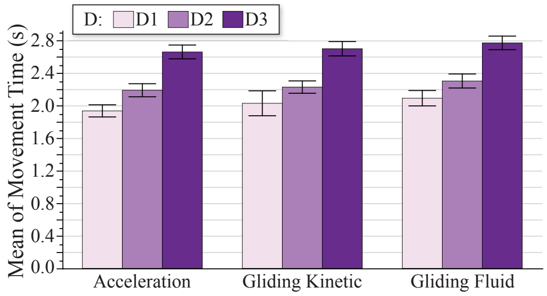

Figure 6: MT by D and by TECHNIQUE. Performance across techniques is similar.

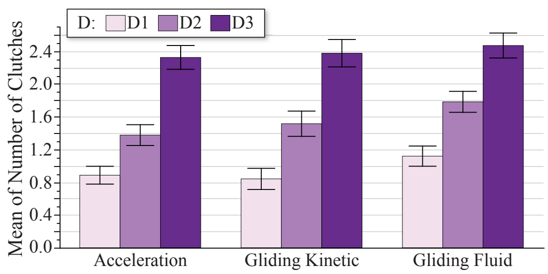

Figure 7: CLUTCHES by D and by TECHNIQUE. Participants clutch in all conditions despite the optimized transfer function.

ing only acceleration, or that they use gliding when available as a substitute to acceleration. We now analyze these two possibilities.

Gliding.

In order to assess the actual use of gliding we compute the binary measure GLIDED as follows: for each trial, if the cursor glided for more than 0.2s during at least one clutching movement then we consider that gliding was used intentionally for this trial[6].

70.8% of trials in the *Gliding* conditions used gliding: 73.3% for *Fluid* friction, 67.8% for *Kinetic* friction (Fig. 8). Participants used gliding almost all the time for the largest distance (89% for *Fluid*, 91% for *Kinetic*). With the smaller distances, they used *Fluid* gliding more often than *Kinetic* gliding: 73% vs. 56% for *D2*, 51% vs. 37% for *D1*. An ANOVA on GLIDED with the model TECHNIQUE×D×W×RANDOM(PARTICIPANT) confirms the significant effect of D ($F_{2,22.31} = 33.70$, $p < 0.0001$) and the significant D×TECHNIQUE interaction ($F_{2,22.08} = 3.51$, $p = 0.0473$).

These results strongly support **H2**: participants used gliding extensively when they could, especially for larger distances. Since they did not have to do so, this implies that they preferred to use gliding rather than just acceleration. We also observed more frequent use of fluid vs. kinetic friction for small and medium distances, consistent with our theoretical analysis.

Cursor Efficiency.

An ANOVA on CE with the model TECHNIQUE×D×W×RANDOM (PARTICIPANT) reveals significant effects of TECHNIQUE ($F_{2,22} = 41.88$, $p < 0.0001$), D ($F_{2,22} = 13.15$, $p = 0.0002$), and a significant D×TECHNIQUE ($F_{4,44} = 10.43$, $p < 0.0001$) interaction effect. For TECHNIQUE, post-hoc Tukey HSD tests show that CE is significantly higher for both gliding techniques (*Fluid*: 0.99, *Kinetic*: 0.97) than for ACCELERATION (avg. 0.86). For D, post-hoc Tukey HSD tests show a significant difference between the small and large distances (avg. *D1*: 0.96, *D3*: 0.92) and between the small and medium ones (avg. *D2*: 0.95, *D3*: 0.92).

[6] 0.2s is the average duration of a clutch in classic pointing [13]

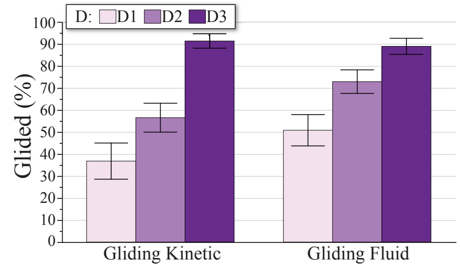

Figure 8: Overall use of Gliding by D and friction type. For shorter distances, gliding is used more often with fluid friction.

For *Acceleration*, the effect of D is significant between all three levels (avg. *D1*: 0.92, *D2*: 0.87, *D3*: 0.80). There is no significant effect of W and D for *Fluid* and *Kinetic*, showing that participants were able to constantly move the cursor even while clutching.

These results strongly support **H3** (Fig. 9). Overall, participants traded the potential efficiency of the acceleration function for gliding, without loss of performance in movement time. This trade-off is captured by cursor efficiency, which is very close to its maximal value in the gliding conditions.

Qualitative results.

The answers to the questionnaire are consistent with the performance results. 10 participants preferred the inertial cursor with fluid friction, 1 preferred kinetic friction, 1 had no preference. All participants said that they mostly used gliding when the target was far away: 8 because it was faster, 7 because it was more comfortable, 6 because it was fun.

10 participants noticed a difference between the two friction types (called A and B in the experiment). 5 participants found kinetic friction more sensitive, with less friction, or with faster acceleration, making it harder to use. Conversely, they found fluid friction to be smother and less sensitive to high velocities. This is consistent with our theoretical analysis: fluid friction is less sensistive to initial velocity, slows down faster and covers a smaller range of distances. We therefore recommend fluid friction for gliding.

6. CONCLUSION AND FUTURE WORK

This article introduces GlideCursor, a pointing technique inspired by the physical metaphor of tossing a puck or rolling a hoop. The goal is to facilitate target acquisition on large displays by letting the cursor continue its course while the user is clutching. GlideCursor works with a relative touch input device such as a trackpad and uses simulated inertia and friction to move the cursor autonomously, based on the movement speed at lift off.

Based on a theoretical analysis, we test several hypotheses on the performance and use of gliding with two studies. We find that glid-

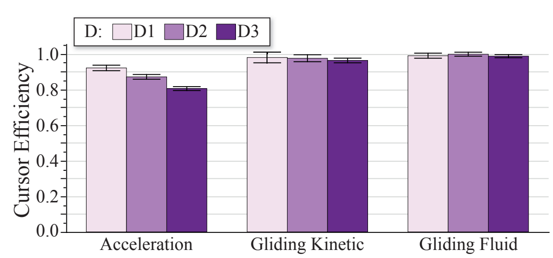

Figure 9: CE by D and by TECHNIQUE. Both gliding techniques are almost optimal, and more efficient than classic acceleration.

ing reduces clutching and can improve pointing performance for large distances. We also find that users spontaneously balance the trade-off between a fast acceleration function and gliding, and use gliding extensively with large distances without degrading performance. We introduce *cursor efficiency* to characterize this effect.

In future work, we plan to study the effects of direction, perform finer analyses of coefficients of friction and combine GlideCursor with other pointing facilitation techniques or with a wider range of input devices. We also plan to develop a predictive model that reflects our understanding of clutching and gliding. Like inertial scrolling, we believe that gliding can become a popular technique that people use without being aware of it, while still delivering quantifiable advantages in performance, comfort and ease of use.

7. REFERENCES

[1] A. Agarawala & R. Balakrishnan. Keepin' it real: pushing the desktop metaphor with physics, piles and the pen. In *Proc. Human Factors in Computing Systems*, CHI '06, p 1283–1292. ACM, 2006.

[2] D. Aliakseyeu, P. Irani, A. Lucero, & S. Subramanian. Multi-flick: an evaluation of flick-based scrolling techniques for pen interfaces. In *Proc. Human Factors in Computing Systems*, CHI '08, p 1689–1698. ACM, 2008.

[3] C. Appert, O. Chapuis, & M. Beaudouin-Lafon. Evaluation of pointing performance on screen edges. In *Proc. Advanced Visual Interfaces*, AVI '08, p 119–126. ACM, 2008.

[4] T. Asano, E. Sharlin, Y. Kitamura, K. Takashima, & F. Kishino. Predictive interaction using the delphian desktop. In *Proc. User Interface Software and Technology*, UIST '05, p 133–141. ACM, 2005.

[5] M. Baglioni, S. Malacria, E. Lecolinet, & Y. Guiard. Flick-and-brake: finger control over inertial/sustained scroll motion. In *Extended Abstracts on Human Factors in Computing Systems*, CHI EA '11, p 2281–2286. ACM, 2011.

[6] R. Balakrishnan. "Beating" Fitts' law: virtual enhancements for pointing facilitation. *Inter. J. of Hum.-Comput. Stu.*, 61(6):857–874, 2004.

[7] P. Baudisch, E. Cutrell, M. Czerwinski, D. C. Robbins, P. Tandler, B. B. Bederson, & A. Zierlinger. Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch- and pen-operated systems. In *Proc. INTERACT '05*, p 57–63, 2003.

[8] M. Beaudouin-Lafon, O. Chapuis, J. Eagan, T. Gjerlufsen, S. Huot, C. Klokmose, W. Mackay, M. Nancel, E. Pietriga, C. Pillias, R. Primet, & J. Wagner. Multi-surface interaction in the WILD room. *IEEE Computer*, 45(4):48–56, 2012.

[9] R. Blanch, Y. Guiard, & M. Beaudouin-Lafon. Semantic pointing: improving target acquisition with control-display ratio adaptation. In *Proc. Human Factors in Computing Systems*, CHI '04, p 519–526. ACM, 2004.

[10] S. K. Card, W. K. English, & B. J. Burr. Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT. *Ergonomics*, 21(8):601–613, 1978.

[11] G. Casiez & D. Vogel. The effect of spring stiffness and control gain with an elastic rate control pointing device. In *Proc. Human Factors in Computing Systems*, CHI '08, p 1709–1718. ACM, 2008.

[12] G. Casiez, D. Vogel, R. Balakrishnan, & A. Cockburn. The impact of control-display gain on user performance in pointing tasks. *Human-Computer Interaction*, 23(3):215–250, 2008.

[13] G. Casiez, D. Vogel, Q. Pan, & C. Chaillou. RubberEdge: reducing clutching by combining position and rate control with elastic feedback. In *Proc. User Interface Software and Technology*, UIST '07, p 129–138. ACM, 2007.

[14] O. Chapuis, J.-B. Labrune, & E. Pietriga. DynaSpot: speed-dependent area cursor. In *Proc. Human Factors in Computing Systems*, CHI '09, p 1391–1400. ACM, 2009.

[15] S. A. Douglas & A. K. Mithal. The effect of reducing homing time on the speed of a finger-controlled isometric pointing device. In *Proc. Human Factors in Computing Systems*, CHI '94, p 411–416. ACM, 1994.

[16] P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *J. Exper. Psych.*, 47:381–391, 1954.

[17] J. Geißler. Shuffle, throw or take it! working efficiently with an interactive wall. In *Extended Abstracts on Human Factors in Computing Systems*, CHI EA '98, p 265–266. ACM, 1998.

[18] T. Grossman & R. Balakrishnan. The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proc. Human Factors in Computing Systems*, CHI '05, p 281–290. ACM, 2005.

[19] Y. Guiard, R. Blanch, & M. Beaudouin-Lafon. Object pointing: a complement to bitmap pointing in GUIs. In *Proc. Graphics Interface*, GI '04, p 9–16, 2004.

[20] M. Hascoët. Throwing models for large displays. In L. Chittaro, editor, *Proc. British Human-Computer Interaction*, volume 2 of *HCI '03*, p 73–77, 2003.

[21] S. Huot, O. Chapuis, & P. Dragicevic. TorusDesktop: pointing via the backdoor is sometimes shorter. In *Proc. Human Factors in Computing Systems*, CHI '11, p 829–838. ACM, 2011.

[22] M. Kobayashi & T. Igarashi. Boomerang: suspendable drag-and-drop interactions based on a throw-and-catch metaphor. In *Proc. User Interface Software and Technology*, UIST '07, p 187–190. ACM, 2007.

[23] I. S. MacKenzie, T. Kauppinen, & M. Silfverberg. Accuracy measures for evaluating computer pointing devices. In *Proc. Human Factors in Computing Systems*, CHI '00, p 9–16. ACM, 2001.

[24] M. McGuffin, P. Dragicevic, & L. Tremblay. Acquiring Targets in the Velocity Domain: Toward Predictive Modeling of Virtual Tossing. Rapport de recherche, May 2012.

[25] D. E. Meyer, R. A. Abrams, S. Kornblum, C. E. Wright, & J. E. K. Smith. Optimality in human motor performance: ideal control of rapid aimed movements. *Psychological Review*, 95:340–370, 1988.

[26] M. Nancel, O. Chapuis, E. Pietriga, X.-D. Yang, P. Irani, & M. Beaudouin-Lafon. High-precision pointing on large wall displays using small handheld devices. In *Proc. Human Factors in Computing Systems*, CHI'13, p 831–840. ACM, 2013.

[27] A. Reetz, C. Gutwin, T. Stach, M. Nacenta, & S. Subramanian. Superflick: a natural and efficient technique for long-distance object placement on digital tables. In *Proc. Graphics Interface*, GI '06, p 163–170, 2006.

[28] S. Zhai. Characterizing computer input with Fitts' law parameters: the information and non-information aspects of pointing. *Int. J. Hum.-Comput. Stud.*, 61(6):791–809, 2004.

[29] S. Zhai & P. Milgram. Quantifying coordination in multiple dof movement and its application to evaluating 6 dof input devices. In *CHI '98*, p 320–327. ACM, 1998.

[30] S. Zhai, C. Morimoto, & S. Ihde. Manual and gaze input cascaded (MAGIC) pointing. In *Proc. Human Factors in Computing Systems*, CHI '99, p 246–253. ACM, 1999.